UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
14/391,686	10/09/2014	Doron Levi	84071857	2229

56436 7590 04/26/2017
Hewlett Packard Enterprise
3404 E. Harmony Road
Mail Stop 79
Fort Collins, CO 80528

EXAMINER	
VU, TUAN A	

ART UNIT	PAPER NUMBER
2193	

NOTIFICATION DATE	DELIVERY MODE
04/26/2017	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

hpe.ip.mail@hpe.com
chris.mania@hpe.com

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

Ex parte DORON LEVI

Appeal 2017-002624
Application 14/391,686
Technology Center 2100

Before JOHN A. JEFFERY, BRUCE R. WINSOR, and
JUSTIN BUSCH, *Administrative Patent Judges*.

JEFFERY, *Administrative Patent Judge*.

DECISION ON APPEAL

Appellant appeals under 35 U.S.C. § 134(a) from the Examiner's decision to reject claims 1–20. We have jurisdiction under 35 U.S.C. § 6(b). We affirm.

STATEMENT OF THE CASE

Appellant's invention tests an integrated software system by (1) intercepting a method call associated with a corresponding object with a mock object including an aspect joined to the corresponding object; and (2) routing the method call to a method associated with the corresponding object or the programmed behavior associated with the mock object. *See generally* Abstract. Claim 1 is illustrative with our emphasis:

1. A testing system for an integrated software system comprising:

> a mock object comprising machine executable instructions on a first non-transitory computer readable medium, *the mock object implemented as an aspect wrapped around a corresponding object within the integrated software system, the mock object to intercept a method call associated with the corresponding object and to determine, based on configuration data, whether to route the method call to the corresponding object or programmed behavior associated with the mock object*; and
>
> a testing agent comprising machine executable instructions on one of the first non-transitory computer readable medium and a second non-transitory computer readable medium, the testing agent comprising the configuration data to instruct the mock object to route the method call to one of the corresponding object and the programmed behavior associated with the mock object.

THE REJECTIONS

The Examiner rejected claim 1 under 35 U.S.C. § 103(a) as unpatentable over Lopian (US 2010/0037100 A1; Feb. 11, 2010), Arcese (US 2013/0007713 A1; Jan. 3, 2013), and Nan (US 2009/0083578 A1; Mar. 26, 2009). Final Act. 2–5.[1]

The Examiner rejected claims 2–11 and 13–20 under 35 U.S.C. § 103(a) as unpatentable over Lopian, Arcese, Nan, Ziegler (US 2012/0084754 A1; Apr. 5, 2012), Braude (US 2011/0239194 A1; Sept. 29,

[1] Throughout this opinion, we refer to (1) the Final Rejection mailed December 7, 2015 ("Final Act."); (2) the Appeal Brief filed April 29, 2016 ("App. Br."); (3) the Examiner's Answer mailed October 13, 2016 ("Ans."); and (4) the Reply Brief filed December 2, 2016 ("Reply Br.").

2011), Gould (US 2012/0173490 A1; July 5, 2012), and Enokido (US 5,933,634; Aug. 3, 1999). Final Act. 5–21.[2]

The Examiner rejected claim 12 under 35 U.S.C. § 103(a) as unpatentable over Lopian, Arcese, Nan, Ziegler, Braude, Gould, Enokido, and Lui (US 2007/0083813 A1; Apr. 12, 2007). Final Act. 21–23.

The Examiner provisionally rejected claims 5 and 10 on the ground of non-statutory obviousness-type double patenting over claims 3 and 8 of Application No. 13/450,788 and Lopian. Final Act. 23–25.

THE OBVIOUSNESS REJECTION OVER LOPIAN, ARCESE, AND NAN

The Examiner finds that Lopian's method for validating software discloses many recited elements of claim 1 including, among other things, a testing system for an integrated software system comprising (1) a mock object, associated with paragraph 12 and Figures 7 and 8, implemented as an aspect "wrapped around" a corresponding real or fake object, associated with paragraph 10, using weaved code coupled to a mock framework, (2) the mock object (a) to intercept a method call associated with the corresponding object and (b) to determine, based on configuration data, whether to route the method call to the corresponding object or programmed behavior associated with the mock object, and (3) a testing *code* responsible for instructing a mock object using the configuration data. Final Act. 2–4.

[2] Although the Examiner omits claims 16–20 from the statement of this rejection, the Examiner nevertheless discusses claims 16–20 in the corresponding body of the rejection. *Compare* Final Act. 5 *with* Final Act. 20–21. Accordingly, we present the correct claim listing here, and deem the Examiner's error in this regard harmless.

Although the Examiner acknowledges Lopian lacks a testing *agent* responsible for instructing the mock object, the Examiner cites Arcese and Nan for teaching the recited testing agent in concluding that the claim would have been obvious. Final Act. 4–5.

Appellant argues that neither Lopian's mock framework nor weaved code is a mock object implemented as an aspect "wrapped around" a corresponding object, as claimed. App. Br. 5–9; Reply Br. 2–14. According to Appellant, Lopian's mock framework is instead *separate* from a production code base, and Lopian's weaved code is instead *embedded* within the production code base. App. Br. 8. Appellant adds that because Lopian's weaved code calls the mock framework that is separate from the weaved code, the weaved code does not intercept a method call associated with a corresponding object and determine, based on configuration data, whether to route the method call to the corresponding object or programmed behavior associated with the weaved code. App. Br. 8–9.

ISSUES

Under § 103, has the Examiner erred by finding that Lopian, Arcese, and Nan collectively would have taught or suggested:

(1) a mock object implemented as an aspect "wrapped around" a corresponding object, giving the term its broadest reasonable interpretation in light of the Specification?

(2) the mock object (a) to intercept a method call associated with the corresponding object and (b) to determine, based on configuration data, whether to route the method call to the corresponding object or programmed behavior associated with the mock object as recited in claim 1?

ANALYSIS

We begin by construing a key disputed limitation of claim 1 which recites, in pertinent part, a mock object implemented as an aspect "wrapped around" a corresponding object within an integrated software system. Claim construction is an issue of law that we review *de novo*. *Cordis Corp. v. Boston Scientific Corp.*, 561 F.3d 1319, 1331 (Fed. Cir. 2009).

The meaning of a claim term may be determined by reviewing a variety of sources including the claims themselves, dictionaries and treatises, and the written description, the drawings, and the prosecution history. *Brookhill-Wilk 1, LLC v. Intuitive Surgical, Inc.*, 334 F.3d 1294, 1298 (Fed. Cir. 2003). Appellant's Specification does not define the term, but does note that a mock object "can be injected into the integrated software system for example, using aspect oriented programming (AOP) frameworks, intermediate language or byte code weaving, or code instrumentation." Spec. ¶ 9; *see also id.* at ¶ 25. Appellant's Abstract describes the mock object as including an aspect "joined" to a corresponding object. Although this description informs our construction of the term, it does not limit our interpretation.

The Examiner and Appellant disagree on the term's construction and provide their own respective interpretations. App. Br. 6–9; Reply Br. 2–9, 12; Ans. 2–7. According to the Examiner, "wrapping around" is an act of substituting or overlapping an object's original function with a wrapper function that results in hiding the object's original function. Ans. 4. According to Appellant, "wrapping around" is an act of embedding or containing a corresponding object's instructions within a mock object's instructions that forms a composite object implemented as instructions on a non-transitory computer readable medium. Reply Br. 7–9, 12.

As is known in the art, a "wrapper" is an object that encapsulates and delegates to another object with the aim of altering the another object's behavior or interface. MICROSOFT COMPUTER DICTIONARY 575 (5th ed. 2002). A "wrapper class" masks a non-object-oriented implementation, hides software components provided by a third-party, and/or encapsulates objects with an interface that is not compatible. DICTIONARY OF COMPUTER SCIENCE, ENGINEERING, & TECHNOLOGY 533 (Phillip A. Laplante ed. 2001). Moreover, to "encapsulate," as included in both the above definitions, is to keep the implementation details of a class in a separate file whose contents need not be known by one using the class. MICROSOFT COMPUTER DICTIONARY 191–92 (5th ed. 2002). Therefore, under its broadest reasonable interpretation, a mock object implemented as an aspect "wrapped around" a corresponding object keeps the implementation details of the corresponding object's class in a separate file with the aim of altering the corresponding object's interface.

Given this construction, we see no error in the Examiner's reliance on the functionality of Lopian's mock version of original code, implemented as an aspect, that is "wrapped around" the corresponding original code's functionality. *See* Ans. 5–7 (citing Lopian ¶¶ 39–46, 57–61, 72, 74, 93–99; Fig. 9); *see also* Final Act. 2 (additionally citing Lopian ¶¶ 10, 12, 15, 30–31, 62; Figs. 1, 3, 7, 8). Notably, Lopian's production code base is code that is to be isolated and tested. Lopian ¶¶ 30–31. Added code weaved into the production code base is shown in Figure 1. The added code allows hooking mock objects into the production code base by calling a mock framework which decides whether to call the original code or fake the call. Lopian ¶¶ 15, 30. Inserting mock objects alters the isolated production code thus suggesting keeping the original code intact in a separate location.

We also see no error in the Examiner's finding that Lopian's mock version of an original code (a) intercepts a method call associated with a corresponding object, and (b) determines, based on configuration data, whether to route the method call to the corresponding object or programmed behavior associated with the weaved code. *See* Ans. 7–8 (citing Lopian ¶¶ 15, 30, 61, 72); *see also* Final Act. 3 (additionally citing Lopian ¶¶ 31–48, 57–60, 62, 65, 94; Figure 6).

Notably, weaver hooks inserted into the base code (Lopian ¶¶ 15, 30, 61, 72) which, as the Examiner explains, are responsible for (a) intercepting method calls of the base code by allowing (b) a real-time determination by the mock framework whether to initiate initial calls or execute fake calls such that the mock call's behavior is changed. Despite Appellant's arguments that Lopian's weaved code calls for a mock framework, (Reply Br. 14), Appellant does not persuasively rebut the Examiner's finding of results from inserting the weaver code into the production code base other than calling a mock framework.

Therefore, we are not persuaded that the Examiner erred in rejecting claim 1.

THE OBVIOUSNESS REJECTION OVER LOPIAN, ARCESE, NAN, ZIEGLER, BRAUDE, GOULD, AND ENOKIDO

Claims 2–7, 9–11, and 13–20

We also sustain the Examiner's obviousness rejection of claims 2–7, 9–11, and 13–20 over Lopian, Arcese, Nan, Ziegler, Braude, Gould, and Enokido.

Appellant argues a *prima facie* case of obviousness has not been established with respect to claim 9 because the Examiner did not explain how Zeigler, Braude, Gould, and Enokido apply to claim 9. App. Br. 10.

We, however, are not persuaded. The Examiner has a duty to give notice of the rejection with sufficient particularity to give Appellant a fair opportunity to respond to that rejection. *See* 35 U.S.C. § 132(a). As the Federal Circuit indicates:

> [A]ll that is required of the [Patent] [O]ffice to meet its prima facie burden of production is to set forth the statutory basis of the rejection and the reference or references relied upon in a sufficiently articulate and informative manner as to meet the notice requirement of [35 U.S.C.] § 132.

In re Jung, 637 F.3d 1356, 1363 (Fed. Cir. 2011).

Here, in the Final Rejection, claims 2, 3, and 9 are indicated as being rejected under 35 U.S.C. § 103(a) as being unpatentable over Lopian, Arcese, Nan, Zeigler, Braude, Gould, and Enokido in the statement of the rejection. Final Act. 5. The Examiner finds Lopian teaches nearly every element of claim 9, but refers to the rationale in claims 2 and 3 for generating a scenario as a hierarchical data object. Final Act. 15. Claims 2 and 3 are directed to a testing system comprising a scenario to store configuration data, where the scenario comprises a hierarchical data structure, and the Examiner relies upon Zeigler, Braude, Gould, and Enokido for teaching those limitations under 35 U.S.C. § 103(a). Final Act. 5–8. The body of the rejection of claim 9 does not explicitly cite Zeigler, Braude, Gould, and Enokido. Nevertheless, given (1) claim 9's hierarchical data object scenario limitation that is commensurate with the scenario recited in claims 2 and 3, and (2) the Examiner's directing Appellant to refer to the rationale in claims 2 and 3 in connection with these commensurate limitations, the Examiner provides sufficient notice to Appellant that the Examiner also relies on Zeigler, Braude, Gould, and Enokido in rejecting claim 9. Therefore, despite the Examiner's somewhat inartful approach in rejecting claim 9, the Examiner's rejection is nonetheless sufficiently articulate and informative to give Appellant a fair opportunity to respond to the rejection.

Accordingly, we find the Examiner satisfied 35 U.S.C. § 132 by setting forth (1) the statutory basis of the rejection of claim 9, and (2) the references relied upon in a sufficiently articulate and informative manner as to meet the notice requirement of 35 U.S.C. § 132. By satisfying the requisite burden of production to justify the rejection of claim 9 under § 132, the Examiner established a *prima facie* case.

Regarding the last clause of claim 9, Appellant further reiterates similar arguments made in connection with independent claim 1 with respect to Lopian's alleged shortcomings in this regard, and alleges that the remaining cited prior art fails to cure those purported deficiencies. App. Br. 10–11. We are not persuaded by these arguments for the reasons previously discussed.

Accordingly, we sustain the rejection of claim 9, and claims 2–7, 10, 11, and 13–20 not argued separately with particularity.

Claim 8

We also sustain the Examiner's obviousness rejection of claim 8, which recites that the testing system further comprises a data binding model in which each of a plurality of mock objects are linked to a property bag that stores data from the mock object.

Appellant does not address—let alone persuasively rebut—the Examiner's interpretation of a property "bag" on pages 13 and 14 of the Answer. In the Answer, the Examiner explains that a property "bag" is a file or memory storage that generates method calls such that a mock object is linked to the property "bag." Ans. 13. The Examiner further explains that a mock object is linked to a property "bag" because the binding information collected in a property bag is integrated into a mock call or method. Ans. 13–14.

Therefore, the Examiner's interpretation has at least a rational basis that has not been persuasively rebutted. On this record, then, the weight of the evidence favors the Examiner's position. Ans. 13–18.

THE OBVIOUSNESS REJECTION OVER LOPIAN, ARCESE, NAN, ZIEGLER, BRAUDE, GOULD , ENOKIDO, AND LUI

We also sustain the Examiner's obviousness rejection of claim 12. Ans. 5–6. Despite nominally arguing these claims separately, Appellant reiterates similar arguments made in connection with the independent claims, and alleges that the additionally cited references fail to cure those purported deficiencies. App. Br. 13. We are not persuaded by these arguments for the reasons previously discussed.

THE PROVISIONAL DOUBLE PATENTING REJECTION

Because Application No. 13/450,788 was abandoned on January 26, 2017, the Examiner's provisional double patenting rejection of claims 5 and 10 (Final Act. 23–25) based on this application is moot and, therefore, not before us.

CONCLUSION

The Examiner did not err in rejecting claims 1–20 under § 103.

DECISION

The Examiner's decision rejecting claims 1–20 is affirmed.

No time period for taking any subsequent action in connection with this appeal may be extended under 37 C.F.R. § 1.136(a)(1)(iv).

AFFIRMED